A Technology & Architecture Integration Project for Product Generation System Design (TAIP)

URL: http://www-ias.jpl.nasa.gov/TAIP/taip.html

JPL Team

(PI)	Meemong Lee	:	System Design Integration
	Mark Abajian	:	PGS Shell
	Alan Mazer	:	Distributed Data Flow System
	Laverne Hall	:	System Architecture/Performance Modeling

Univ. Minnesota

	Vipin Kumar	:	Processor Benchmarking Task Scheduling

Collaboration

	Graham Bothwell	:	MISR PGS prototyping

Planned Activities (TAIP)

PGS Shell Application

- MISR PGS (L1-L2) Beta Software
- TES/ MODIS PGS Software

PGS Simulation

- Function Descriptor Validation
- Function Emulation
- Performance Model Emulation

Technology Evaluation Metric

- Design Trade-Off Analysis
- Evolvable Software Design Guideline

Objectives (TAIP)

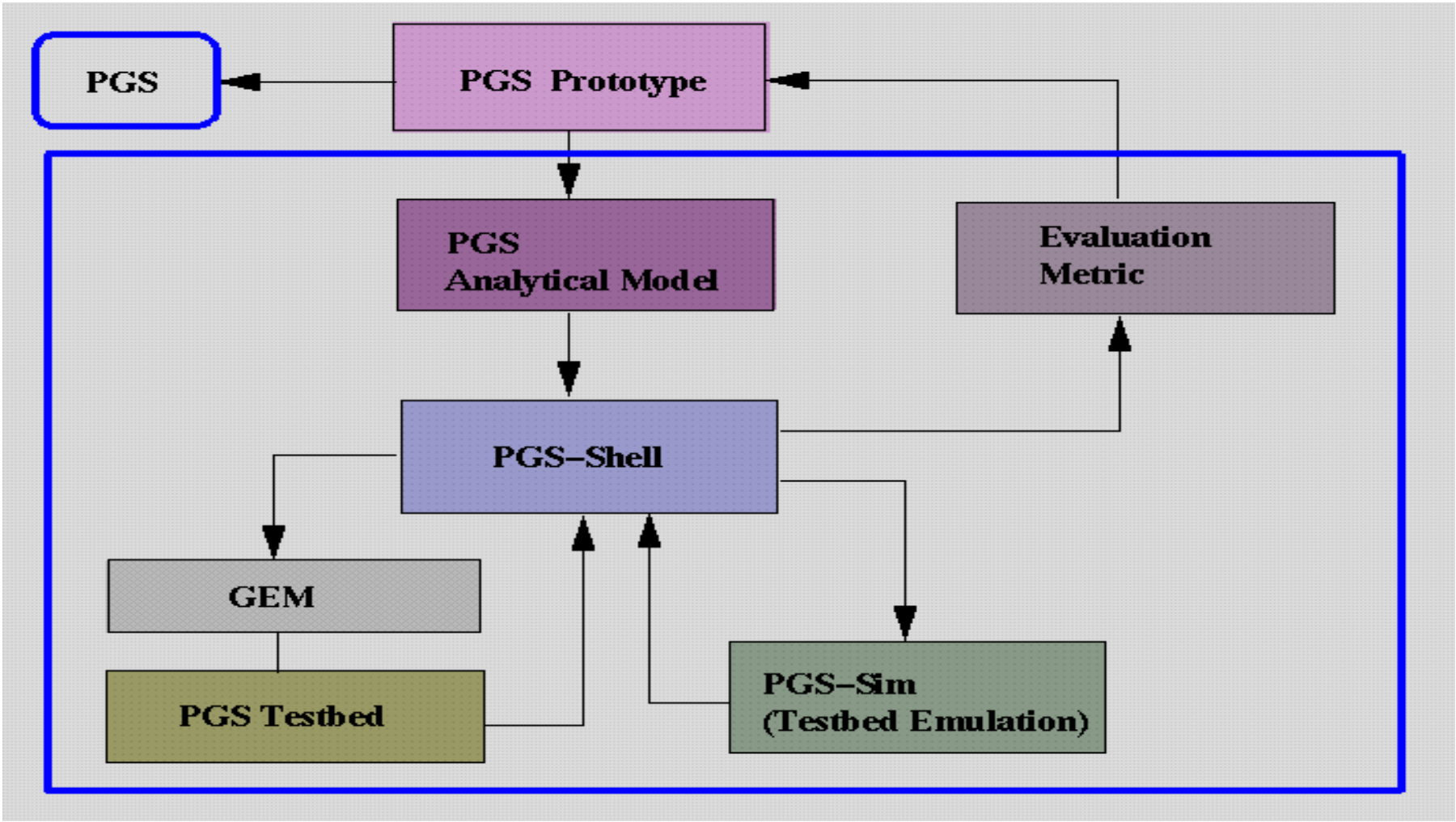
EOSDIS Product Generation System Model

PGS Architecture Prototyping

Evovable System Design Environment

Technology Evaluation Metric

PGS Design Environment (TAIP)



Keywords (TAIP)

Function - Software Implementation Unit

Process - Software Execution Unit within which functions share the address space (A set of Functions)

Product - EOSDIS archival Data set

Process Module - Product generation procedure Unit (A set of Processes)

Processor - Hardware Unit (CPU + Memory + File system)

Platform - Product Generation System (A set of Processors)

Data Flow - Inter-process data traffic

Process Flow - Process Execution Sequence

Shared Object (.so file) - Dynamically Loadable Object Code containing a set of Functions

EOSDIS PGS MODEL (TAIP)

Approach:

A PGS Descriptor Design for expressing

- Product-oriented Process Execution Control Flow

-Inter-Process Communication using PGS-Specific Data Structures

- Computation & I/O Characteristics of Processing Functions

Status:

PGS model descriptor Implementation

-Process Module Descriptors

-Platform Descriptors

PGS Architecture Prototyping (TAIP)

Approach:

Heterogeneous Distributed Computing Testbed
- Heterogeneous Operating Systems
- Distributed Memory Architecture
- Shared Memory Architecture

PGS Case Study
- High-Fidelity (On-platform
- Architecture Performance Modeling
- Resource Utilization Modeling

Status:

PGS Testbed Configuration at JPL and Univ. MN

PGS Architecture Prototyping (TAIP)

Approach:

PGS design experiment environment which provides

- Dynamic Process Configuration
- Process Mapping to Heterogeneous Distributed Processors
- Interactive Process Monitoring and Benchmarking

Status:

GEM V 2.5 completion

PGS Shell V 1.0 completion

MISR L1B2 Prototype implementation on PGS Testbed

Technology Evaluation Metric (TAIP)

Approach:

Technology Impact Analysis Environment Design

- Computational Resource Utilization Model in terms of Relevant Technology Components (CPU, I/O storage)
- PGS Platform Emulation
- PGS Software Emulation

Status:

Network II.5 based PGS Testbed Configuration

Power Challenge System Architecture Model Implementation

MISR L1B2 Process Model Implementation

Rude Awakening (TAIP)

PGS Software Schedule Change

Software System Complexity

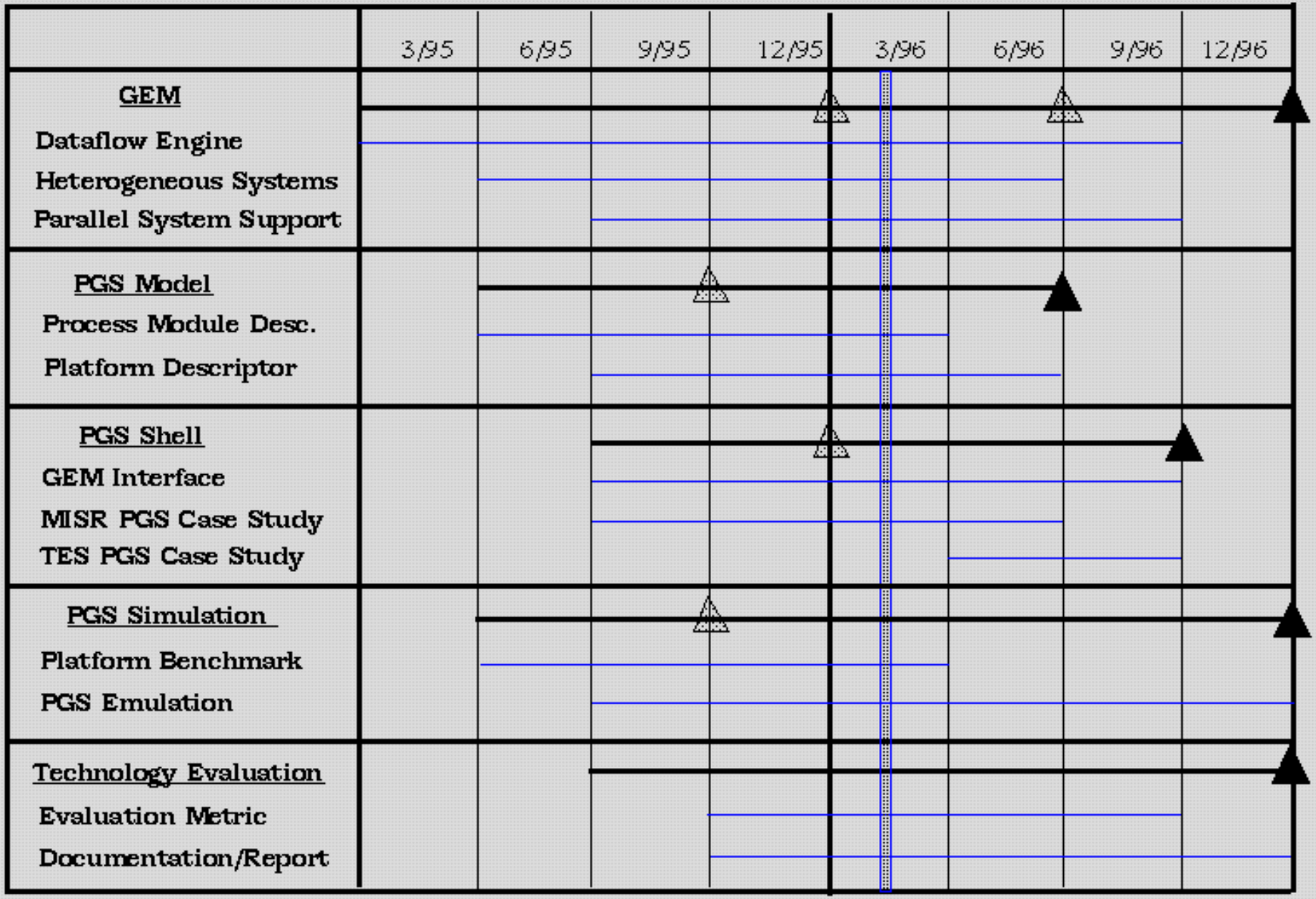
Software Licensing

Testbed Platform Availability

Government Shutdown

JPL

Revised Schedule (TAIP)



Deliverables (TAIP)

GEM V 2.5

Demonstration: Multi-Platform Distributed Computing

Supported Platforms: SUN, HP, Alpha, SGI, SGI-Power Challenge

Capabilities

- Graphical Dataflow Interpretation
- Automated Process Loading & Execution
- Automated Inter-Process/Processor Data Transportation
- Interactive Data Product Visualization
- Performance Monitoring

Limitations

- Multi-Level Process Control Flow
- Feedback Loop
- Heterogeneous Datatype Structures

Deliverables (TAIP)

PGS Shell V 1.0

Demonstration: MISR L1B2 PGS Beta Software

Supported Platforms: SUN, Indigo II, Power Challenge

Capabilities

- Automated GEM Interface from PGS Descriptors
- Application Data Structure Handling
- Interactive User Input Broadcasting
- Embedded Dataflow Control (Looping)

Limitations

- Automated Process to Processor Mapping
- Transfer of Data referenced by a Pointer across the process boundary